

PCI Bus Prototyping Card (I)

Explore the PCI-Bus

By H. Kolter (Kolter Electronic)

The development of a PCI based expansion card is relatively complex and usually needs a large investment of time and money before success is achieved. The PCI prototyping card presented here offers a more convenient solution. Its pre-configured PCI decoder and 16 bit wide data bus together with a series of port addresses allows you to concentrate on development and testing of your prototype circuit without getting bogged down in the detail of the PCI bus standard.

The Peripheral Component Interconnect or PCI Bus is a relative newcomer to the computer interface world but has rapidly replaced the older and well known ISA Bus. One attractive feature for peripheral designers is that this bus is used not only for IBM compatible PCs but also for workstations and Apple Macintosh systems. Basically, the interface is a very powerful multiplexed 32/64 bit wide

address/data bus. One of the original aims of this bus standard was to ensure that it would not become obsolete overnight and would still be in use after many generations of processor have come and gone.

The PCI bus is independent of the system processor and is almost a stand-alone system bus capable of a relatively high data transfer rate.

Modern PC motherboards use PCI slots and no longer provide any of the older ISA slots. The disadvantage for the designer of a PCI plug-in card is that the design process is a bit more complex. It is no longer possible to fiddle with hardware jumpers to select the address space of the expansion card — instead, the card configuration is performed with software. The PCI card contains its own operating system that is initialised by writing to internal registers over the PCI interface. The PCI BIOS will recognise each of the PCI

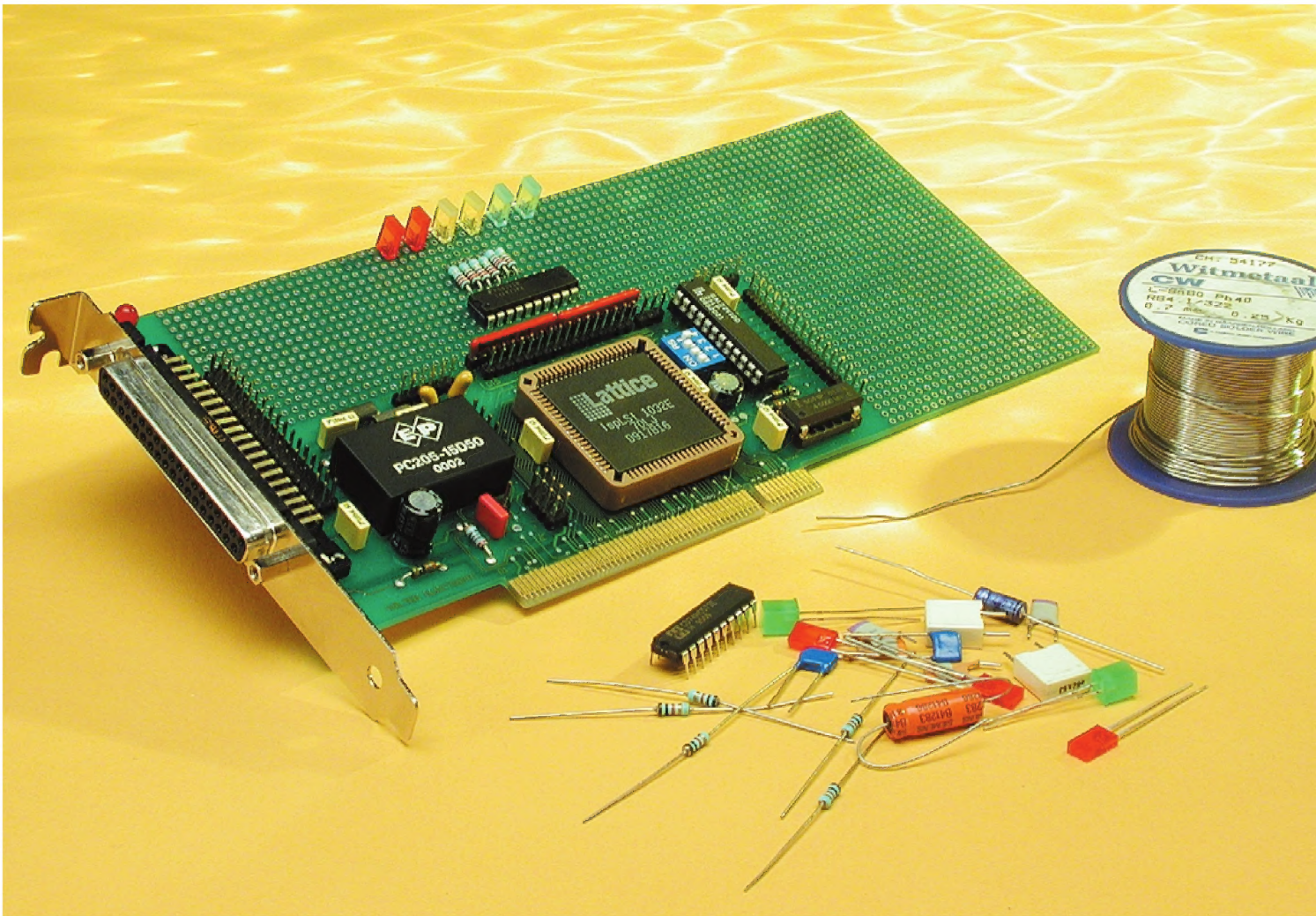
expansion cards fitted to the computer system and will initialise them during the BIOS set-up procedure. Once the card has been configured under BIOS, it is assigned an access address. There are different PCI routines that allow this information to be read or altered.

PCI Controller and Vendor ID

The central core of the PCI expansion card is the PCI decoder or controller chip. This chip forms the link between the PCI bus of the computer and your custom circuitry. Most commercial PCI controller chips on the market are not only very complex (and expensive) but are also relatively unwieldy beasts because of their large package outlines. For our PCI experimentation card in this project, the PCI controller has been implemented in an ispLSI1032 pro-

PCI board features:

- 32-bit PCI Controller implemented with ispLSI
- 16 bit User Interface similar to ISA-Bus
- Accessible Decoder GAL22V10 (Synario)
- Large 2.54-mm matrix of through-plated holes
- DC/DC converter +/- 15 volts
- Suitable for prototyping digital or analogue circuits
- Selectable Product ID
- 8/16 bit Bus with CS signals



grammable chip from Lattice. This solution is not optimal for every PCI application — it was developed to be used for current 16 bit ISA applications. The decoder is intended to be used for the slow and middle speed data rate of the PCI specification less than or equal to 33 MHz, supporting the specifications 2.1 and 2.2 of the PCISIG group of the PC98 and PC99 specification. The PCISIG group is also responsible for assigning a unique Vendor Identity (VID) code to each manufacturer of PCI compatible products and also a Device ID code for each type of PCI product that the manufacturer produces. These codes, along with information of the cards I/O and memory address range and interrupts, are contained in a 64-byte information header stored on the card. This information is read by the PCI BIOS whenever the computer is re-booted.

Technical Data

Digital Output:
Modes:

PCI Decoder:

Vendor-ID:

Product-ID:

Addressing:

Bus:

User Interface:

Connectors:

37 pin sub-D connector (K4), connector on slot bracket

Pin strips for the breadboard wiring:

1 x 9 pin. (K9), ± 15 V, AGND and +5 V, GND, power supply for analogue prototyping components.

1 x 6 pin. (K8), interrupt select

2 x 17 pin (K5), patch pins for the output socket. (see Figures 1 and 5)

1 x 3 pin (K6), not defined. Useful for application-specific prototyping signals.

1 x 6 pin (K10), 3 way power supply +5 V, GND, for prototyping digital circuits.

1 x 16 pin (K3), data bus

1 x 16 pin (K2), I/O-Bus

Dimensions:

95 x 215 mm (card dimensions without bracket)

Temperature range

Storage temperature

1 x TTL level (0/+5 V)

TTL OUT port polling,
or program controlled

1 x ispLSI 1032E (Lattice)

KOLTER 0x1001

KOLTER 0x0017 (adjustable)

variable, PNP (Plug and Play)

32/64 bit PCI

8/16 bit

0-70 °C typical. (continuous operation)

-20 to +85 °C

Card features

The PCI prototyping card provides an area of through-plated holes of 87×37 solder points (minus 46×21 for the PCI decoder). This breadboard area provides a useful space to lay out prototype hardware rapidly and easily for your specific PCI application, e.g., A/D or D/A converters, TTL I/O circuitry or general-purpose interfaces. A power supply providing ± 15 V from a separate DC/DC converter is also included on the card and a 16 bit data bus together with a series of port addresses ensures that you will be able to access your hardware in under 90 ns. All the necessary components are provided on the card and configured to transfer data to and from the PCI bus.

Configuration of the card address occurs automatically with Plug and Play (PNP). Extensive PCI tools running under Windows 95/98/NT and 2000 enable the address I/O address range of the card to be determined. No further initialisation of the components is necessary to be able to read and write to the card at register level.

The circuit

A quick look at the block diagram in **Figure 1** shows that the PCI prototyping card consists of four main parts:

IC1: PCI bus adapter chip type ISPLSI1032

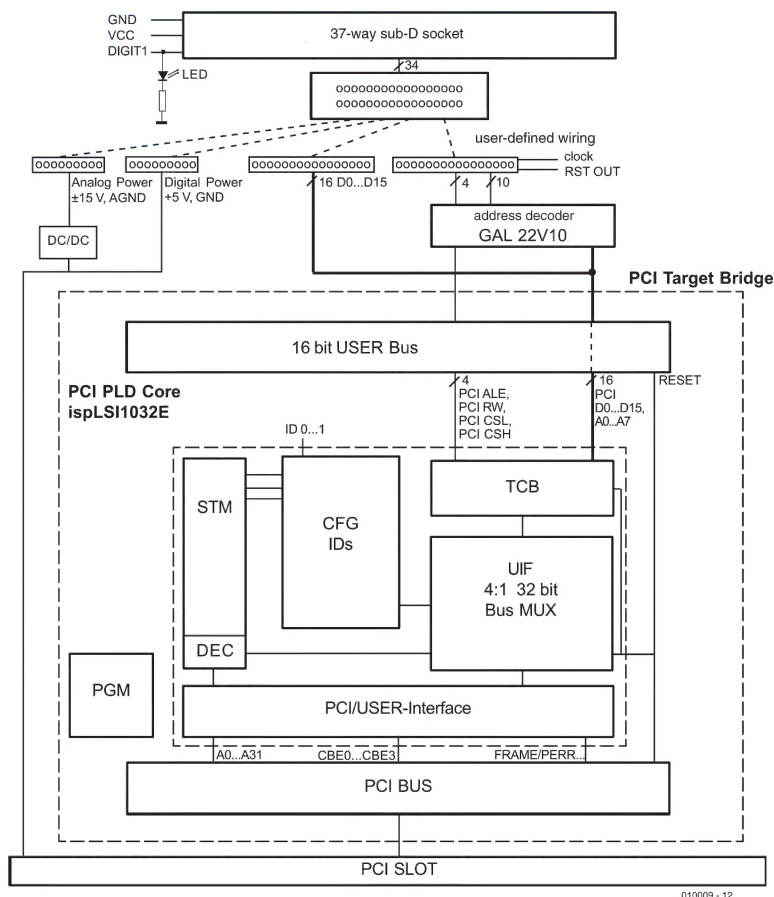


Figure 1. Overview of the functional blocks of the PCI prototyping card. An explanation of the abbreviations used is given in the inset.

Abbreviations used in the Block Diagram

PCI/USER Interface: Bus interface with bi-directional port and bus driver for the signals: CLK, System Pin, Address/Data Pins, CBE and other Interface-Control Pins along with Error Reporting Pins.

STM, DEC and CFG-ID: STM stands for State Machine, this reads out the information to the PCI bus over the 4:1 32-bit multiplexer of all the 64 bytes of CFG registers when the information request is made by the PCI BIOS, e.g., during computer boot-up. The CFG register is 'pre-wired'. The product ID inputs ID0, ID1 and ID2 allow selection of one of eight ID's. The Vendor ID is pre-programmed and not selectable.

UIF and Bus-MUX: Bus driver and Turnaround are generated here for the PCI target. The bus multiplexer co-ordinates the address and data in both directions. All the read and write instructions are used in this interface.

TCB: The Timing Correction Block generates pre-defined Bus Cycles so that the user bus runs with precise timing. This block is necessary to cater for motherboards with different bus timings.

PGM: Programming Interface for the ispLSI chip (not freely accessible).

ALE: Address Latch Enable, switches the multiplexed user bus between address and data.

D0 - D15: 16-bit wide data bus. Data is latched with ALE and CSL/CSH.

A2 - A7: Address lines used to generate the memory mapped chip selects. Address line A0 is not used (decoded from CSL and CSH). Address line A1 is also not used. Only the lower 16 bits of the 32-bit data word can be decoded.

RW: User Bus Signal Read/Write: for decoding the read and write cycles for the I/O timing

CSL / CSH: Chip Select Low and Chip Select High signal. When an 8-bit input or output is performed CSL will be active when the lower 8 bits of the 16 bit data word are used and CSH will be active when the upper 8 bits are used. If a 16-bit instruction is used then both lines are active so that a 16-bit word can be read or written in one cycle. It is always the lower 16 bits of a 32-bit data word that are decoded.

RESET: Activating this line will force the chip into its initialisation configuration.

with ID switch.

IC2: address decoder gate array type GAL 22V10.

IC4: DC/DC voltage converter chip together with its Radio Fre-

quency Interference (RFI) suppression components.

K2 to K10: these rows of pin strips and connectors are free to be wired to the prototyping area of the card.

K4 is a sub D type connector providing connections to external signals.

Looking (ahead) at the circuit diagram (**Figure 2**) and the PCB layout in **Figure 3** you'll

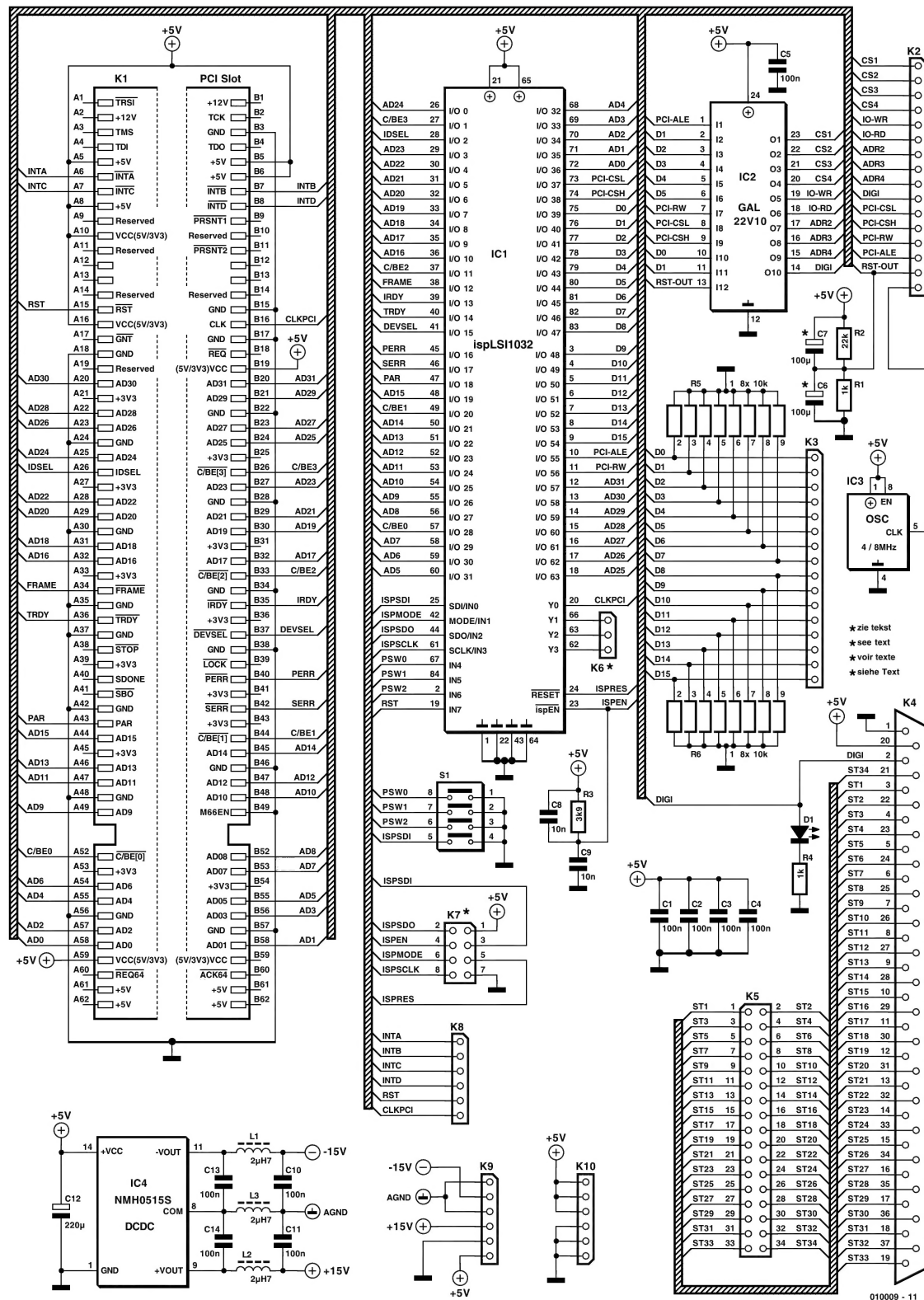


Figure 2. Circuit diagram of the PCI prototyping card. The biggest part of the circuit, the PCI bridge which links the PCI bus to the prototyping area circuits, is implemented in the ispLSI-Chip.

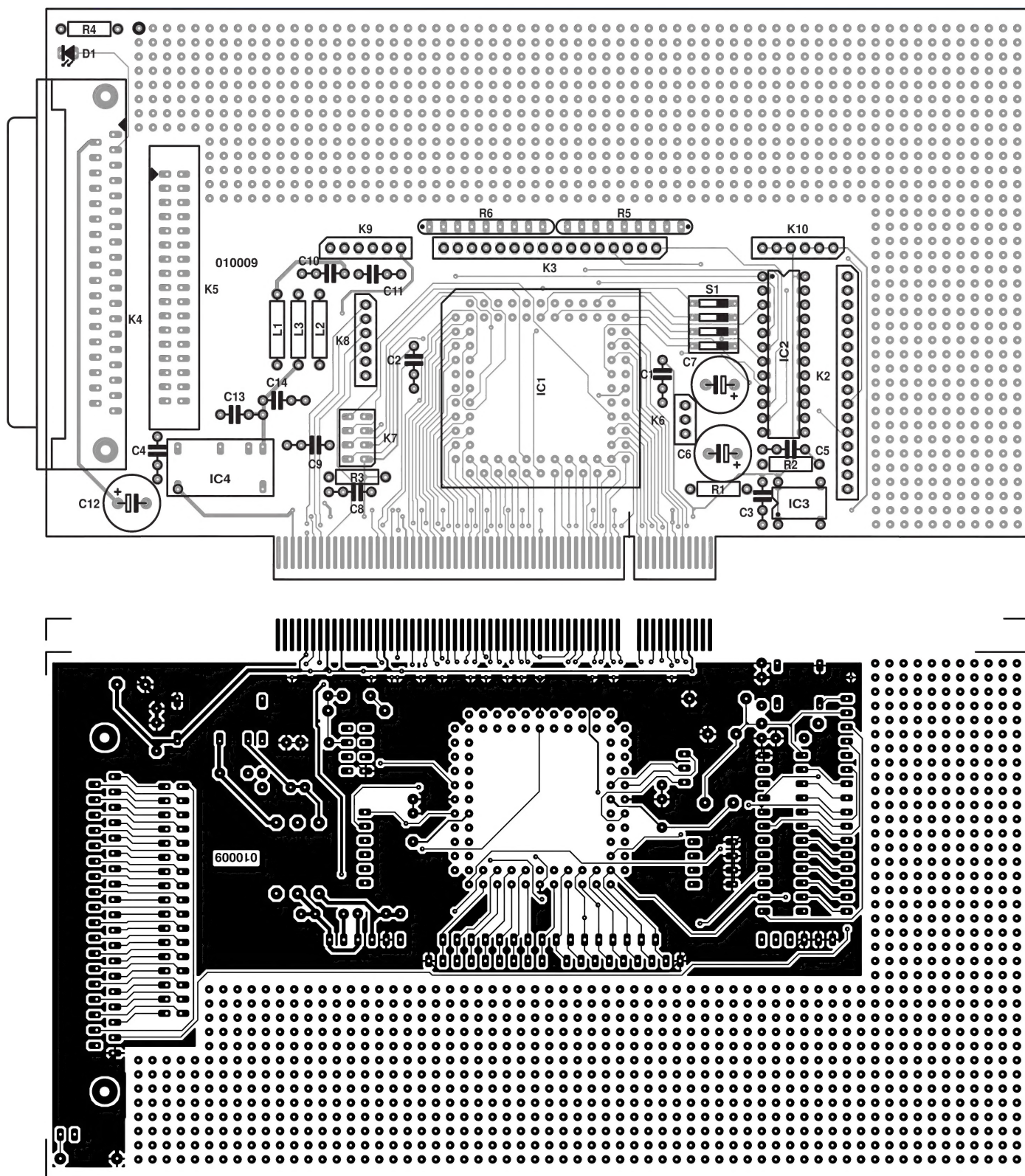


Figure 3. PCB layout and component placement plan.

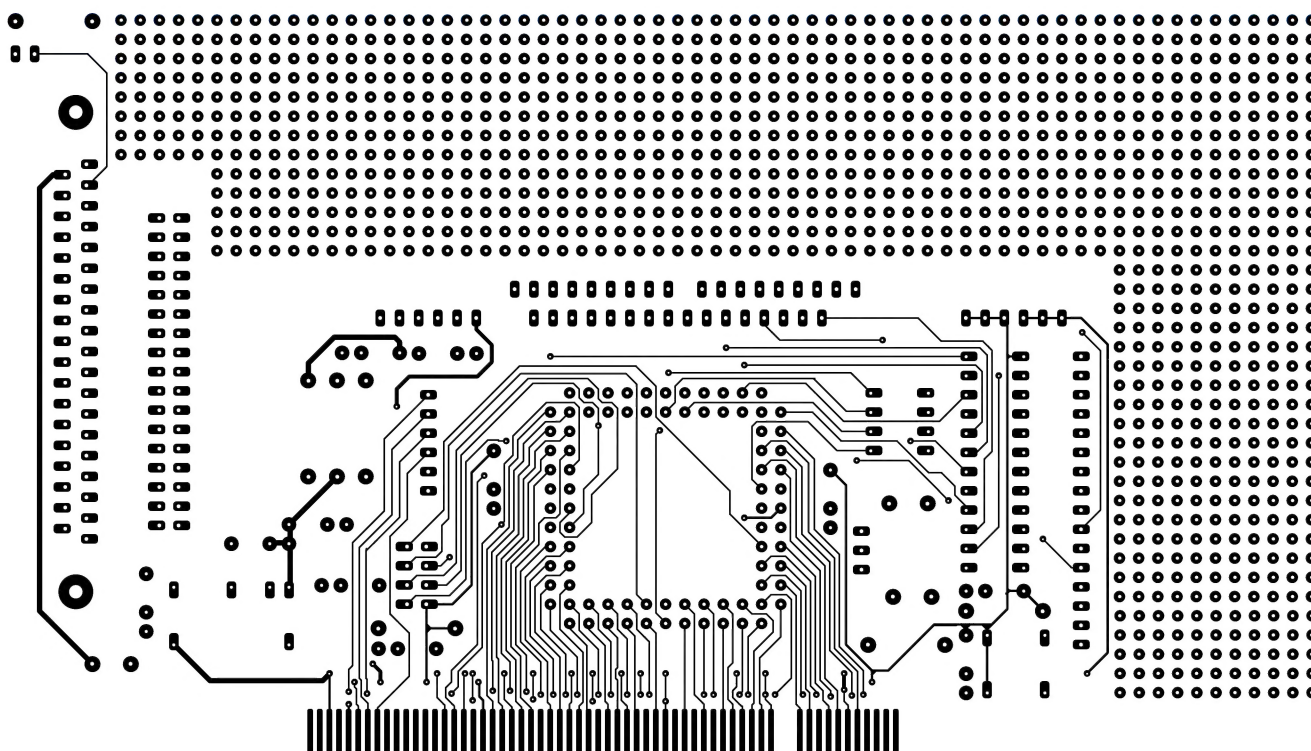
notice that the most complex function of the circuit is performed by IC1, the pre-programmed LSI chip ispLSI1032 from Lattice.

The block diagram for the present PCI card is shown in **Figure 1**. IC1 is shown here as

the 'PCI target bridge'. A key to the abbreviations used on this diagram is given at the end of the text.

The Vendor ID code mentioned earlier is pre-programmed into IC1

and in this case has the value 0x1001. This is the VID of Kolter Electronic, Germany, and can be used on all the applications in which the ispLSI-chip from Kolter is employed.



COMPONENTS LIST

Resistors:

R1, R4 = 1k Ω
 R2 = 22k Ω
 R3 = 3k Ω
 R5, R6 = array 8 x 10k Ω

Capacitors:

C1-C5, C10, C11, C13, C14 = 100nF
 C6, C7 = 100 μ F 16V
 C8, C9 = 10nF
 C12 = 220 μ F 16V

Inductors:

L1, L2, L3 = 2 μ H7 (see text)

Semiconductors:

D1 = LED
 IC1 = ispLSI1032 (order code
010009-41)*
 IC2 = GAL22V10 (order code

010009-42)*

IC3 = oscillator module in 8-pin DIL case (see text)

IC4 = NMH0515S (see text)

Miscellaneous:

K1 = PCB edge connector for PCI bus
 K2, K3 = 16-pin SIL pinheader
 K4 = 37-pin sub-D socket
 K5 = 34-pin boxheader or pinheader
 K6 = not fitted**
 K7 = not fitted**
 K8, K9, K10 = 6-pin pinheader
 S1 = 4-way DIP switch
 PCB, order code **010009-1***

* For programmed controllers, GALs and PCB, see Readers Services pages elsewhere in this issue, or website at <http://www.elektor-electronics.co.uk>.

** See text

IC1 forms the bridge between the computer PCI bus and the 16-bit wide bus of this prototyping card. If your application only requires an 8 bit bus then it is a simple matter to

leave the other 8 bus bits unused. The data transfer rate will be halved, but this is not always an important criterion and may be perfectly adequate for your application. Expand-

ing the bus to 32 bits wide is unfortunately not possible with this LSI.

DIP switch S1 has two functions. The fourth of the four switches, S1-4, allows the switching between the PCI local specification 2.1 and 2.2 while switches S1-1, S1-2 and S1-3 allow the device ID to be changed (**Figure 4**). This allows several PCI prototyping cards to be plugged into the same PCI bus, and the control software will be able to distinguish between each card on the basis of its vendor ID, device ID and its slot number.

IC2 is a programmed gate array containing an address decoder which produces four chip selects outputs CS1, CS2, CS3 and CS4 and generates a RD and WR signal along with three decoded address lines ADR2, ADR3 and ADR4. A 1-bit digital output also provides an indication that the circuit is working correctly by its output to LED D1 via resistor R4. It is important to note here that when chip select $\overline{CS1}$ and $\overline{CS3}$ are active, data transfer occurs over the data bus pins D0 to D7. When $\overline{CS2}$ and $\overline{CS4}$ are active, data transfer occurs over D8 to D15. When a 16-bit transfer occurs, $\overline{CS1}$, CS3 and CS2, CS4 are active.

The programming data for the GAL is freely available and is shown in **Figure 5**. The GAL listing is also given in one of the files that can be downloaded from the Free Downloads section of the *Elektor Electronics* website at <http://www.elektor-electronics.co.uk>. Using this information it is possible to change

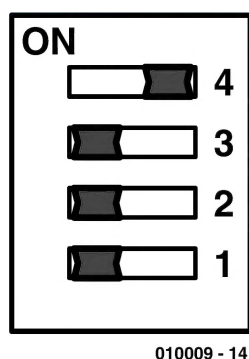


Figure 4. Selecting the Product ID with the DIL switch (example shown 17 hex).

the chip selects or generate new I/O signals for your own applications or special processor control signals.

The programming tools for the Lattice GAL are available from the Internet. If you intend to redesign the address decoder, it should be noted that half of the 256 addresses are used because we only use 16 of the 32 bits available. All addresses with address bit A1 set to a '1' are invalid.

Additional features

In addition to the basic ICs necessary to perform the PCI interface, there are also some pre-wired IC positions on the PCB to accept additional ICs that may be useful for prototyping.

Components R1, R2 and C6, C7 can be fitted to build a reset circuit in one of two variants: either active-Low or active-High reset. Fitting R1, R2 and C7 will produce an active-High reset signal at the RST-OUT pin while fitting R1, R2 and C6 will produce an active-Low reset signal. In this case, R2 is 1 k Ω and R2, 22 k Ω . It should be noted that either C6 or C7 should be fitted but not both.

The position for IC3 will accept a clock generator chip and that for IC4, a DC/DC converter type NMH 0512S or NMH 0515S for generating either ± 12 V or ± 15 V respectively. These supply voltages are the most useful when prototyping with analogue devices.

Capacitors C10 to C14 and inductors L1 and L2 will filter out any nasty supply spikes.

Placing jumpers across the pins of K7 will allow the 'In Circuit Programming' mode to be enabled of the LSI IC1. This is primarily intended as an aid for the development of the LSI. Those of you who are not familiar with these features of this LSI are advised to leave K7 unused, otherwise you run the risk that the PCI card may no longer function.

Building your own circuits

When constructing your own circuits on the breadboard area of the card it is possible to use IC's from the 74ALS, 74ACT, 74AHC or 74F families. The 74HCT family of devices has a switching speed that is just on the limit of acceptability when used on the PCI card, and reliable operation of the hardware cannot be guaranteed. A test circuit was built using HCT devices in the *Elektor Electronics* laboratory, and when these chips were swapped with HCT devices from a different manufacturer, the circuit failed to work. In the interests of reliability, it is wise not to include HCT devices in your designs built on this prototyping card.

If you are interfacing to any of the PCI bus signals, it is important to note that the specification for this bus allows a maximum of one TTL load on each of the PCI bus signals with a maximum track capacitance of 10 pF. This means that apart from the PCI decoder chip it is not possible to make any further connections to the PCI bus connector. If the PCI decoder chip IC1 is used in another development circuit, it is vital that the track layout to this chip is the same as in this PCI prototyping card. Changes to the track layout or track length can cause timing problems in the PCI decoder chip. The 16-bit data bus is provided for circuit expansion.

The Software Driver

The driver for the PCI prototyping card can be downloaded from the Free Downloads section, February 2001 items, on the *Elektor Electronics* internet site <http://www.elektor-electronics.co.uk>. Along with the dri-

vers are also example programs together with their source code running on DOS, Windows 9x, Windows 2000/NT and Linux. This software is also available on a CD-ROM called 'PC Card Software CD' from Kolter Electronics. Also included on the CD are shareware programs, demos, datasheets, PCI card documentation and useful Internet links.

Use with DOS

For operation of the PCI prototype card on a computer using DOS, it is not necessary to add any drivers in the CONFIG.SYS or AUTOEXEC.BAT files. DOS software can directly access the hardware using the assembler instructions IN and OUT.

PCVIEW

This DOS program displays all the installed PCI bus components together with information showing each Vendor ID and Device ID. It also shows (in as far as possible) the manufacturer's name and a description of all devices used on the card. A more important feature of this program is that it shows the base address of the card connected to the PCI bus. This information allows you to determine the I/O address of the card and use this value in your own software to read and write to the PCI card.

One important point to watch here is that the address allocation of the card may be altered automatically if there is a change in the PCI system, e.g., installing new components or removal of hardware. In that case, you will also need to change the corresponding address in your software.

Using the PCI card base address in your program is the most direct method of programming the card, although it means that the resulting program will not be portable, that is, it will only run in this one environment and not on another PC or if changes are made to the hardware configuration. To make the program portable it is necessary to make use of the PCI functions of the PC BIOS because DOS does not have PCI Applications Programming Interface, API.

PCI_ADR

This program is similar to PCVIEW

Table 1

Outputs of the address decoder implemented in GAL IC2

CS1	Base + 0
CS2	Base + 1
CS3	Base + 4
CS4	Base + 5

but only shows information of PCI devices with the Kolter Electronics vendor ID (including this PCI prototyping card).

Use with Windows 95/98/NT

The installation of the PCI prototyping card is just as simple as with any other PCI cards under Windows 95/98. After fitting the card in the PC it will be recognised by Windows when the PC is turned on. You will be presented with the message that the new PCI card has been detected and then asked which driver to install. You now need to refer to the file `KOLTER.INF` (see the chapter on drivers). Windows now looks for the `KLIBNDRV.VXD` file and installs the driver. When the computer is restarted the PCI prototyping card will be installed.

HWT

The Hardware Test Program HWT is a PCI diagnosis program running under windows. Besides the manufacturer's vendor ID and the equipment device ID it also shows the I/O base address of the card. The displayed base address has been incremented by one so that for example a address shown as E001 is actually E000. The address space of the PCI card extends from this base address over a 256-byte area. In our example, the actual address range would be from E000 to E0FF. Using the base address, it is possible to access the ports using 8- or 16-bit data width.

The functions `Read*106` and `Write*106` are included and can be used to determine the maximum data flow through the port of your computer. Clicking on this function will perform 1000000 Port cycles. Measuring the time taken will give the throughput of your system. The hardware table is a standard text file and can be easily expanded to your requirements.

With Windows 95/98/ME you don't need any further registration to use HWT — all the data files can be stored in any subdirectory. If you are using Windows NT4/2000 then it is necessary to copy SYS data into the directory `c:\Winnt\system32\drivers` and then re-boot the system. HWT

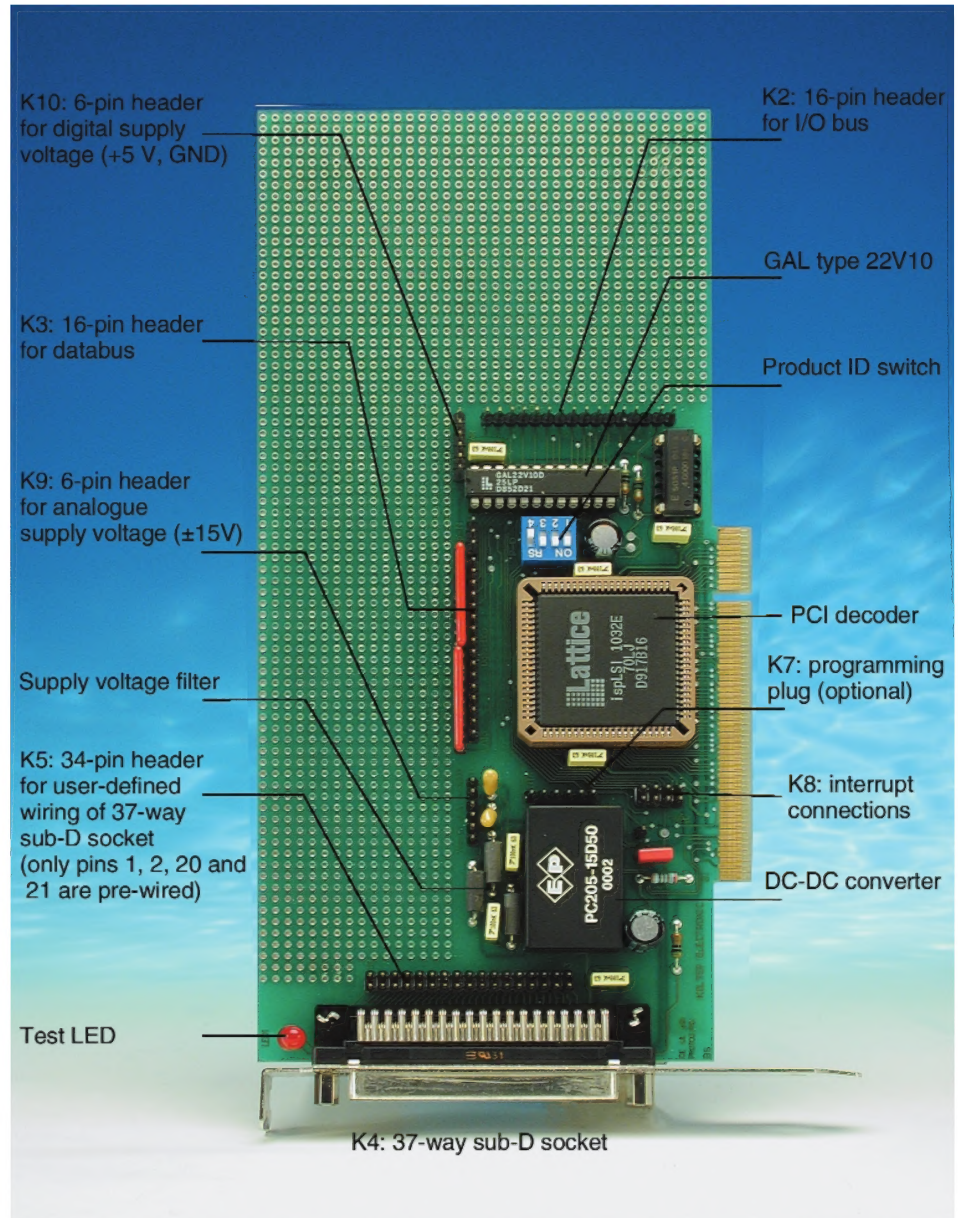


Figure 5. Overview of the pinheaders and main elements on the board.

can then be launched from Explorer.

Example Program

This program, complete with the Delphi source code, allows you to rapidly test your own circuitry in the breadboard area of the PCI prototyping card. It also allows you to see if the PCI card itself is functioning correctly by flashing the test LED on the card.

An NT driver is available for Windows 2000/NT. Details are available at the web address <http://www.pci-card.com>. (see web links at the end of this article).

Use with Linux

Users of the Linux operating system will find it generally very simple to program hardware cards such as I/O cards, A/D and D/A converters using the GNU-C compiler. Comparing it with the MS operating system you will find that there are no hidden access mechanisms or tricks that you normally find necessary with MS operating system programming. The Linux operating system and Linux application software is openly available and more often than not free. Programming in Linux is so much simpler that anyone familiar with ANSI-C should have no problems. The instructions `ioperm` and `iopl` give the program permission to access the I/O ports without the need for any additional device

drivers or the need for any extra cryptic include statements in the compilation process. The programming process is similar to ANSI-C running in DOS but instead of only an 8-bit wide data access you can also use 16 or 32 bit. To actually move data through the ports, instructions such as `outb(value, address)` and `inb(address)` are contained in the directory `/usr/include/asm/io.h`. These routines are all inline macros so it is only necessary to `#include<asm/io.h>`. No additional libraries are required. Compilation of the program can be performed with XWPE under KDE or with gcc from the Linux 'bash' shell. Other compilers can also be used to compile the code and they generally seem to suffer fewer problems than when under other operating systems. Extensive information on the use of the Linux operating system is available as a download from the *Elektor Electronics* website.

Data transfer, timing and addressing

The second instalment of this article will explore in detail the high speed data transfer, timing and addressing of the PCI bus architecture, it will also look at the BIOS con-

figuration and the PCI bus connector pin assignments in 5 V and 3.3 V systems.

(010009-1)

Suggested literature:

PCI Bus Demystified, by Doug Abbott, LLH Technology Publishing, ISBN 1-878707-54-x.

Supplier reference:

Kolter Electronic,
PO Box 1127,
D-50362 Erftstadt,
Germany.
Tel. (+49) 2235 76707,
Fax (+49) 2235 72048.

Web Links

Links for the PCI prototyping card and PCI decoder:

<http://www.pci-card.com/pcintern.htm>
<http://www.pci-card.com/neuep.htm#proto3>
<http://www.pci-card.com/Prototim.pdf>
<http://www.pci-card.com/pcideko.pdf>
<http://www.pci-card.com/pcideko.html>
<http://www.pci-card.com/pciversion.html>
<http://www.pci-card.com/pciboards.html>
<http://www.pci-card.com/Hwt.zip>
http://www.pci-card.com/pci_pins.htm
<http://www.pci-card.com/pro3pci.zip>

